*Journal of Applied and Natural Science*
13(4), 1339 - 1349 (2021)
ISSN : 0974-9411 (Print), 2231-5209 (Online)
journals.ansfoundation.org

*Research Article*

# Artificial intelligence-powered expert system model for identifying fall armyworm infestation in maize (*Zea mays* L.)

**R. Prabha**
Department of Agricultural Entomology, Agricultural College and Research Institute, Tamil Nadu Agricultural University, Coimbatore - 641003 (Tamil Nadu), India
**J. S. Kennedy***
Department of Agricultural Entomology, Agricultural College and Research Institute, Tamil Nadu Agricultural University, Coimbatore - 641003 (Tamil Nadu), India
**G. Vanitha**
Department of Physical Science and Information Technology, Agricultural College and Research Institute, Tamil Nadu Agricultural University, Coimbatore - 641003 (Tamil Nadu), India
**N. Sathiah**
Department of Agricultural Entomology, Agricultural College and Research Institute, Tamil Nadu Agricultural University, Coimbatore - 641003 (Tamil Nadu) India
**M. Banu Priya**
Department of Agricultural and Information Technology, Agricultural College and Research Institute, Tamil Nadu Agricultural University, Coimbatore - 641003 (Tamil Nadu) India

*Corresponding author. Email: jskennedy@tnau.ac.in

***Article Info***
https://doi.org/10.31018/jans.v13i4.3040
Received: September 24, 2021
Revised: November 7, 2021
Accepted: November 12, 2021

***How to Cite***
Prabha, R. *et al.* (2021). Artificial intelligence-powered expert system model for identifying fall armyworm infestation in maize (*Zea mays* L.). *Journal of Applied and Natural Science*, 13(4), 1339 - 1349. https://doi.org/10.31018/jans.v13i4.3040

**Abstract**
Maize (*Zea mays* L) is one of the most saleable cereal crops grown worldwide and a dominant staple food in many developing countries. The severe outbreak of fall armyworm in maize causes massive yield loss. Modern technologies, including smartphones, can assist in detecting recognising the fall armyworm infestation in maize. The objective of this study was to develop an automated Artificial Intelligence Powered Expert System (AIPES) for identifying fall armyworm infestation in maize. In addition, it put forward a deep learning-based model that is trained on photographs of healthy and fall armyworm infested leaves, cobs and tassels from a dataset and furnished an application that will be detecting maize fall armyworm infestation using Convolutional Neural Network (CNN) architecture and Mobile Net V 2 framework model. The study developed an Artificial Intelligence (AI) based maize fall armyworm infestation detection system using a DCNN (Deep Convolutional Neural Network) to support maize cultivating farmers. The model executed the objective by accurately identifying the fall armyworm infested maize plant and also classified them vis-c-vis the healthier crop. The deep learning models were trained to detect and recognise fall armyworm infection using more than 11000 images of fall armyworm infested leaves, cobs, and tassels. The created application (AIPES for identifying fall armyworm infestation in maize) using CNN detected and recognised the fall armyworm infestation in maize with a 100 per cent training accuracy rate and 87 per cent validation accuracy. So, the detection of maize fall armyworm and the treatment of fall armyworm-infested maize could lead to a higher maize crop yield.

**Keywords**: Artificial intelligence, Convolutional neural networks, Deep learning, Fall armyworm infestation detection, Maize, Mobile app

## INTRODUCTION

Maize has emerged as a crucial cereal crop in peninsular India, where states like Andhra Pradesh, which ranks fifth in terms of area (0.79 m ha), have recorded the highest production (4.14 m t) and productivity (5.26 t ha-1) in the country, despite the fact that productivity in some of Andhra Pradesh's districts is more or equal to that of the United States (Murdia *et al*., 2016). In the recent past, maize crop has been imperilled by the fall armyworm (*Spodoptera frugiperda*, J.E. Smith) (FAW) that has caused substantial yield loss in maize. Fall armyworm is native to tropical and subtropical America and is acknowledged as a sporadic pest in the United States since 1797. The spread of fall FAW in India has been proclaimed since it is documented in Karnataka in

May 2018 (Sharanabasappa *et al*., 2018). Automatic diagnosis of plant diseases from captured images through computer vision and artificial intelligence research is feasible to technological advancements (Wang *et al*., 2013; Cheng *et al*., 2017). This research proposes the expert system that will enable to detect fall armyworm infestation in the field using the new technology of artificial intelligence to identify and classify the fall armyworm infestation in the maize crop.

Image processing techniques are now widely employed in agriculture and it is concerned with the detection and recognition of infestations of plants. Computer vision and image processing technology are widely engaged in a variety of industries, and they have a wide range of applications in modern agriculture (Wang *et al*., 2013). Compared to the visual method of detecting plant disease, automatic detection proceeds less time and effort and is more accurate (Singh and Misrab, 2016). Machine learning technologies afford numerous opportunities to promote an automatic disease classification. However, these consume a lot of time manually extracting the features from the images and feeding them as input to the algorithm to classify the plant disease (Huddar *et al*., 2012).

Currently, more deep learning models have been delineated to be gaining popularity than others. Deep learning is a novel technology for image processing and object recognition that improves the accuracy of crop disease classification. Transfer learning is a prominent deep learning approach in which pre-trained models are converted to perform a new task (Cheng *et al*., 2017; Nanni *et al*., 2020; Alves *et al*., 2020). A Convolutional Neural Network (CNN) is a deep learning model widely used in image processing and simply detects and categorises the images through its multi-layered structure. CNNs are at the heart of most cutting-edge computer vision solutions for a spacious range of applications (Szegedy *et al*., 2016). Deep learning has a wide range of applications, including computer vision, image categorization, restoration, speech, and video analysis**.** Since 2015, image categorization research has leaned heavily on deep learning (Barbedo *et al*., 2018). In their study, Kawasaki *et al.* (2015) and Kulkarni (2018) illustrated a unique plant disease detection system based on convolutional neural networks. CNN can procure great classification performance using only training photos. Deep convolutional networks were employed to innovate a plant disease recognition model based on leaf image categorization (Sladojevic *et al*., 2016). In the realm of image classification, convolutional neural networks (CNN) have rendered excellent results. Yao *et al*. (2017) used image processing to advocate a three-layer detection approach for detecting and identifying White-backed planthoppers. For the identification of distinct developmental stages of planthoppers on rice plants, the proposed method was proven to be viable and successful. it is effective at evaluating graphical images and extracting the relevant features. The present work aimed to study a convolutional neural network's probability of classifying fall armyworm infested plant leaves, cobs, and tassels images taken directly from maize research plots of Tamil Nadu Agricultural University (TNAU) and eight blocks in Tiruvannamalai districts. Fall armyworm infested leaves, cobs and tassels were classified utilising the pretrained Mobile Net $V_2$ framework which was adopted for building the neural network. Furthermore, the model was deployed in android mobile.

## MATERIALS AND METHODS

### Architecture of image processing system

The present system of Image processing system comprised of five phases namely as image acquisition, image pre-processing, data augmentation, feature extraction and classification as used by Akhtar *et al.* (2013).

### Image acquisition

A large dataset was used to classify the maize fall armyworm infested leaves, cobs, tassels. Maize FAW infested leaves, cobs and tassels dataset of JPEG images was shot with a Nikon D7500 P-Digital Camera. A total of more than 11000 photographs with varying severity levels of infestations were shot from FAW infested maize field and stored in the system for image processing by following the methodology described (Kulkarni, 2018; Militante, 2019; Syarief and Setiawan , 2020). Dataset was collected from various maize growing research plots of TNAU and various blocks (Perunthuraipattu, Thenmudiyanur, Agarampalipattu, Allapanur, Rayandapuram, Kankayanur, Varakur and Perumanam) of Tiruvannamalai districts of Tamil Nadu. Then the collected images were annotated based on the symptoms caused by the FAW (Table 1).

### Image classification used for model development

Maize tassel was also classified as “Healthy tassel” and “Fall armyworm infested tassel” (Fig. k-l). The image dataset in maize is shown in Fig.1. A leaf was identified and deemed as a “Healthy maize leaf” when it remained entirely green (Fig.1a). However, if there were pinholes on the leaves, that could be considered as the “Pinhole symptom” caused by fall armyworm (Fig.1b). “Circular hole symptom caused by the FAW” can be identified if there were any circular holes on the leaves (Fig.1c). If there were elongated lesions on the leaves, it could be considered as the “Elongated lesion-symptom” caused by fall armyworm (Fig.1d). If the whorl leaf had any fall armyworm infestation, it was considered as “Whorl leaf eaten by FAW symptom” (Fig. 1e). Maize cob was classified based on the per cent infestation of fall armyworm (Fig.1f-j).

**Table 1.** Description and total number of image dataset

| Maize Leaves | |
|---|---|
| **Classes** | **Number of photos taken** |
| Healthy leaves | 1000 |
| Pinhole caused by FAW | 1963 |
| Circular hole caused by FAW | 2271 |
| Small to several lesions caused by FAW | 3374 |
| Whorl leaf is eaten by FAW | 1000 |
| Maize cob | |
| Classes | Number of photos taken |
| Nil damage to slight damage at tips of the cobs | 196 |
| < 25 % of cob area showing FAW infestation | 384 |
| 26 - 50 % of cob area showing FAW infestation | 151 |
| 51 – 75% of cob area showing FAW infestation | 124 |
| > 75 % of cob area showing FAW infestation | 238 |
| Maize tassel | |
| Classes | Number of photos taken |
| Healthy tassel | 100 |
| FAW infested tassel | 459 |

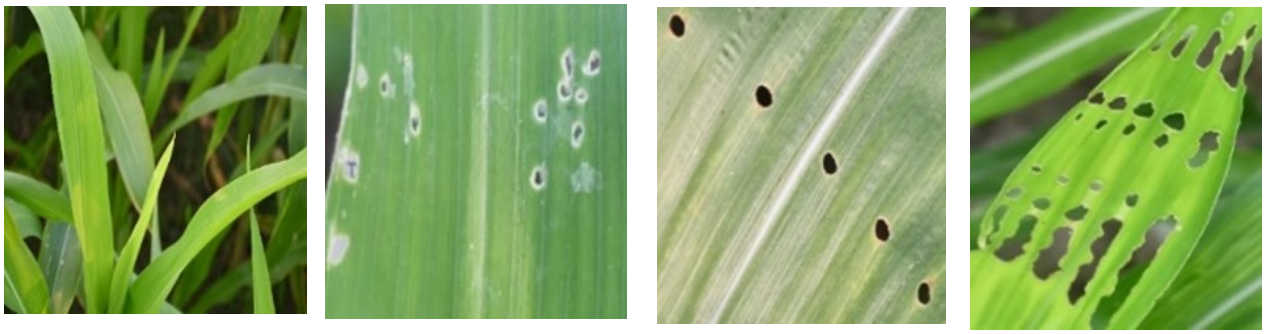


**Fig.(1a)** **Fig.(1b)** **Fig.(1c)** **Fig.(1d)**

**Fig.(1e)** **Fig.(1f)** **Fig.(1g)** **Fig.(1h)**

**Fig.(1i)** **Fig.(1j)** **Fig.(1k)** **Fig.(1l)**

**Fig.1 a-l.** *Showing Images of the dataset in maize*

**Image pre-processing**

Images were reduced in size and cropped to fit a certain input. It improves and processes the image to the required colour scale. Image pre-processing is done for reducing noise and segmenting the image, which boosts the accuracy of the CNN model. The acquired images are usually messy and un-normalized since it comes from different sources. To feed them to the algorithm, they need to be pre-processed, normalized and cleaned up. More often, image pre-processing and normalization is done to cut down the complexity and strengthen the accuracy of the applied algorithm by generating accurate results.

**Data augmentation**

When training a deep learning model, data augmentation is necessary to avoid the overfit problem produced by a small dataset. Supplemented the dataset acquired in the image acquisition to increase the dataset size and offer some variation (Random rotation, random flip, random brightness, random zoom) in image distortion. Augmentor, a Python image augmentation package for machine learning, is used in the data augmentation process.

**Feature extraction and image analysis**

Image analysis entails breaking down an image into its constituent parts to retrieve useful data. Image analysis includes finding shapes, recognising edges, eliminating noise, counting objects, and computing texture analysis or image quality. Feature extraction is a step in the feature dimensionality reduction process, which divides and reduces a large amount of raw data into smaller groupings to make processing easier. Convolution and pooling layers of Convolutional Neural Networks (CNN) are used to accomplish this process.

**Image classification**

Fully connected layers of Convolutional neural network (CNN) are used for the classification of image datasets, whereas convolutional and pooling layers are used for feature extraction. The categorization technique classifies plant portions infested with fall armyworm

**Background knowledge**

**Convolutional Neural Network (CNN)**

The CNN is a sort of artificial neural network that is used for image identification (the capacity to recognise objects) and processing pixel data for a specific design. It consists of an input layer, many hidden layers, and an output layer in the CNN architecture. In the hidden layer, it has a convolutional layer, a Rectified Linear Unit (ReLu), a pooling layer, and a fully connected layer.

A nominal process of a convolutional neural network can easily encounter and categorize the images. Through its multi-layered structure, it is compelling at evaluating graphical images and extracting the relevant features. The architecture of the CNN model was depicted in Fig.2.

**Convolution layer**

The feature map of the input image is extracted in convolution layer through a series of mathematical algorithms. The convolution layer is used to reduce the size of a 6x6 image input into a 4x4 output image matrix using 3x3 filter. 'Convolved feature' refers to the output matrix obtained by sliding the filter over the image from the upper left corner of the input image and computing the dot product. The results are the added values for each step multiplied by the filter's values. The input image is converted into a new matrix with a smaller size (4x4). Fig. 3 depicts the convolution operation.

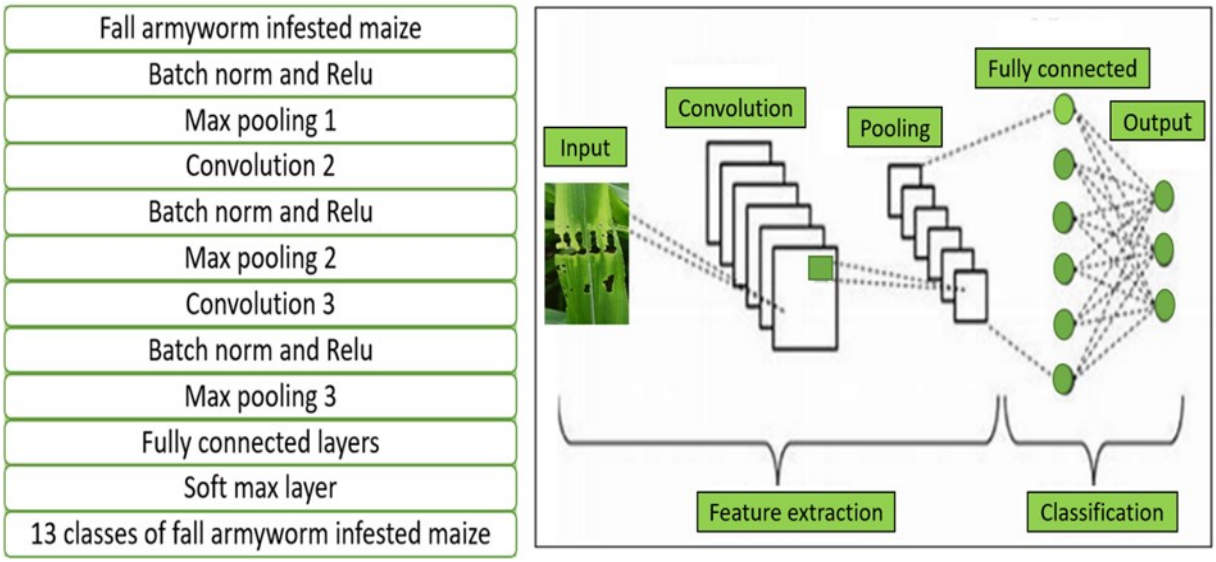



**Fig. 2.** *Architecture of convolutional neural network*

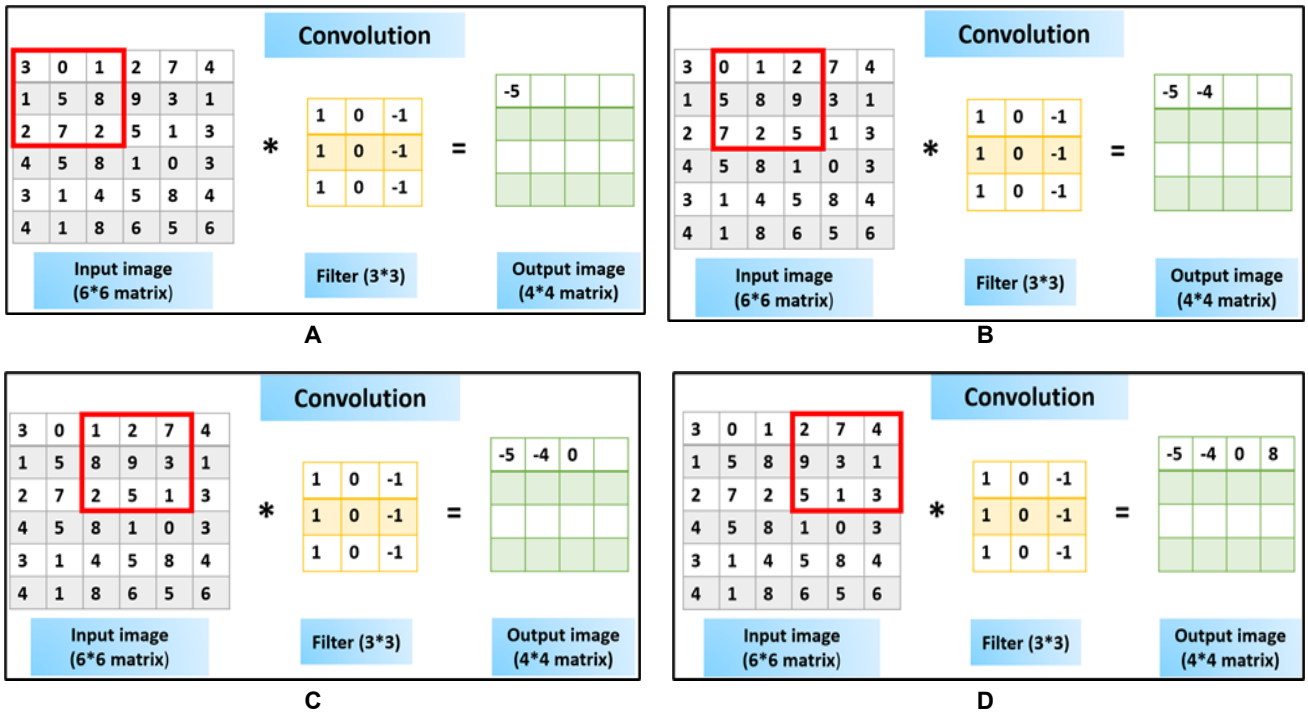



**Fig. 3.** *Convolution operation of input image data A) First mathematical algorithm convolution operation B) Second mathematical algorithm convolution operation C) Third mathematical algorithm convolution operation D) Fourth mathematical algorithm convolution operation*

**Padding**

Padding is used to preserve the original dimensions of the input. In this step, zeros are added to outside of the input image. Numbers of zero layers depend upon the size of the kernel. Padding operation was depicted in Fig.4.

**Pooling layers**

For the down sampling layer, this layer reduces overfitting and shrinks the neuron size. Pooling operation in action was depicted in Fig.5. This layer minimises the size of the feature map, as well as the number of parameters, training time, computation rate, and overfitting. A model is said to overfit if it achieves 100% accuracy on the training dataset and 50% accuracy on the test data. To reduce the size of the feature map, ReLU and max-pooling were used.

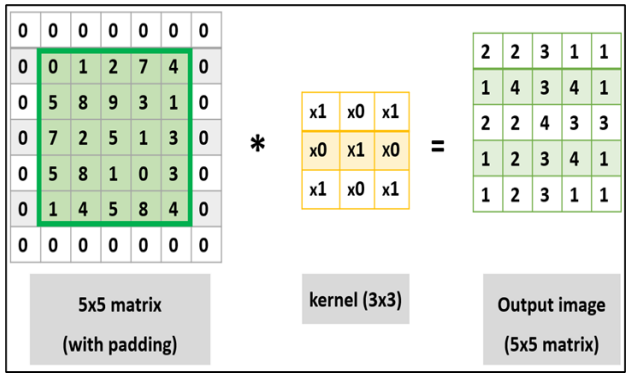



**Fig. 4.** *Padding operation*

**Activation layer**

Every convolution layer makes use of a non-linear ReLU (Rectified Linear Unit) activation layer. Dropout layers are also used in this layer to prevent overfitting.

**Fully connected layers**

This layer employs the SoftMax classifier, a well-known input classifier, to recognise and categorise maize fall armyworm infestation.

**Train, validation and test dataset**

In this study, the dataset of fall armyworm infested maize was split into 70 per cent, 20 per cent and 10 per cent for training, validation and testing, respectively.

**Software and hardware system**

The list of hardware and software used in this study are depicted in Table 2. In the Google Colaboratory editing platform, python 3 was exerted for algorithm implementation and data wrangling scripts. TensorFlow, a sophisticated framework built by Google, was practised to train and infer models. This library supported both CPU and GPU training and inference.

**Google Colaboratory**

Google Colaboratory is a free cloud service granted by google to anybody with a Gmail account and allows anyone to evolve and execute Python code in their browser. People who don't have enough resources can use Google Colab to acquire a GPU for research. The

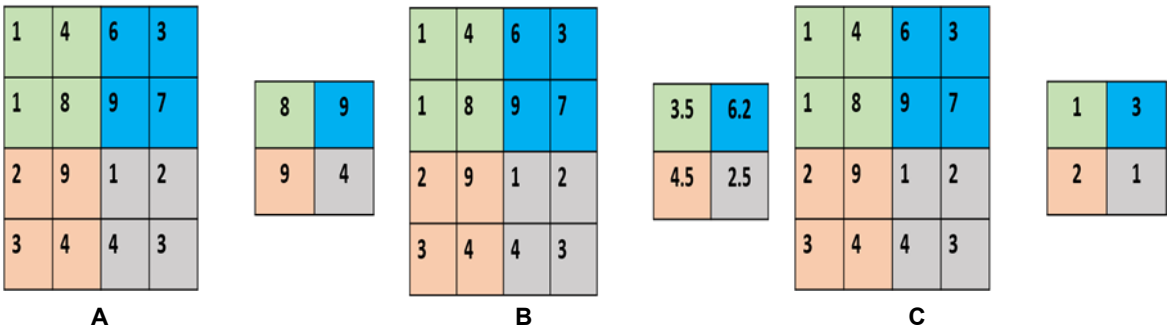



**Fig. 5.** *Pooling operations. A. Max pooling operation. B. Average pooling operation. C. Min pooling operation*

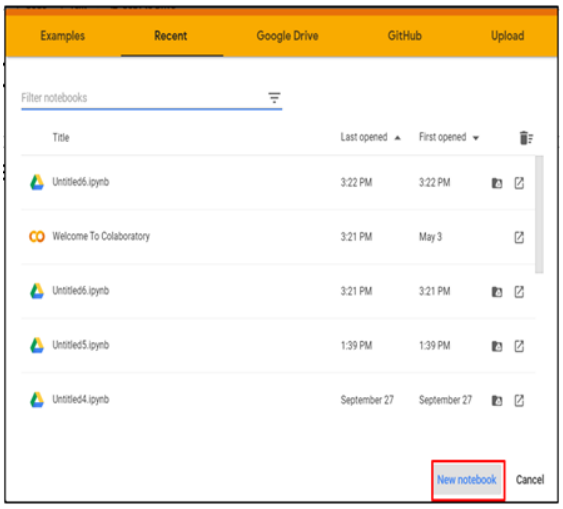



**Fig.6.** *New notebook in Google colab*

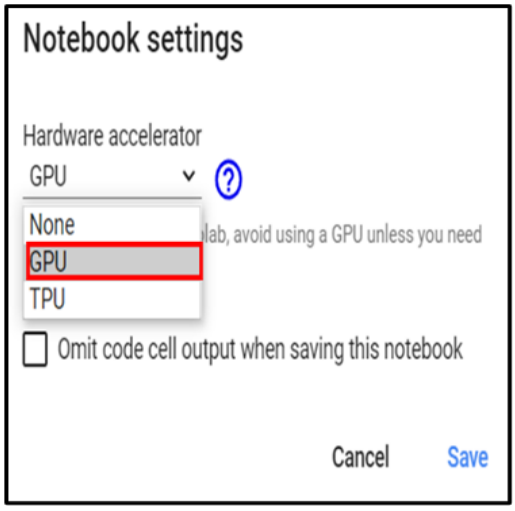



**Fig. 7.** *Change runtime in notebook*

Google Colab service delivers 12.72 GB of RAM and 358.27 GB of hard disc space in one playtime. Every runtime lasts for 12 hours before being reset and the user has to re-establish a connection. This is to hinder anyone from using the GPU service to mine cryptocurrency. A new notebook was built where all Python coding was used to train the model (Fig. 6). The selection box was positioned in Runtime -> Change runtime type and appears as shown below (Fig. 7). The file was connected to a runtime. This connect button was at the top right corner of the page, as shown in the illustration below (Fig. 8).

To run a cell, Ctrl + Enter or click on the cell was used. The play button on the far left of each cell, as indicated in fig.9. The code snippet below (Fig.10) was used to link Google Colab to Google drive. The user was bestowed with a URL and a text box when the code snippet was run. The user clicked on the URL and provided Google Colab permission to access our Google Drive and its data. Once permission was granted, the user was given a one-time token that must be input into the textbox below the URL.

Google Drive is contentedly mounted onto Google Colab if the token is detected and authenticated. Once

**Table 2.** Hardware and software used in this study

| Hardware and Software | Specification |
|---|---|
| Memory | 8.00 GB (5.88 GB usable) |
| Processor | AMD Ryzen 5, 3500U with Radeon Vega Gfx 2.10 GHz |
| Operating system | Windows 10 |
| Editor platform | Google colaboratory |
| Deep learning libraries | Keras and TensorFlow |
| Deep learning architecture | Convolutional neural network |
| Deep learning model | Mobile Net V2 |

Google Drive is mounted onto Google Colab, working code into the cell could be written.

**Working with libraries**

TensorFlow is an open-source machine learning library for research that is fast, versatile and scalable. TensorFlow Lite and TensorFlow Serving, which offers the same features for mobile platforms and high-performance servers that allow to construct and train ML models on computers and mobile devices and servers. Keras is a popular open-source python neural network and it was hastily integrated into TensorFlow's core library, allowing it to be used on top of the framework. Neural layers such as activation and cost functions, goals, batch normalisation, dropout, and pooling are a few of Keras's building blocks and techniques for constructing a neural network.

**Steps involved in training an image dataset with Google Colaboratory**

**Uploading image dataset**

A Gmail account is required to submit image datasets and uploaded all of the image datasets to Google Drive in their original image sizes.

**Splitting image dataset in google drive**

The image datasets were divided into train and validation datasets once the uploading is completed. Two folders entitled train and validation dataset were established in the maize dataset folder.

**Import libraries in google colaboratory**

After the uploading process is complete, create a Google collaborative workspace and write Python code to train the model A few key Python libraries were imported to get started.

**Creating base dataset**

Set the base dataset and add extra layers to the model after installing the Python libraries.

**Generating model summary**

The model summary was then generated, which includes numerous layers, parameters, and gradients based on the given base dataset.

**Print the layers**

The layers were printed once the model summary has been generated.

**Mounting google drive**

Using the Google authorisation code, there is a need to mount Google Drive. So, all of the files were uploaded using this method to train the deep learning model.

**Creating training and validation dataset directory**

For both training and validation, provide a directory and set goal size, colour mode, bath size, class mode, and subset.

**Compile the model**

The model was constructed by setting the number of epochs, decay rate, learning rate, and Adam optimizer after generating the train and validation directories.

**Call back set for training the model**

A callback is a function that is passed as an argument to another function. From Keras, import the model check point and tensor board.call back.

**Training the model**

Training the dataset was done after configuring the call back function. On 10 and 100 epochs, the model was trained. As a result, training the model will take a long

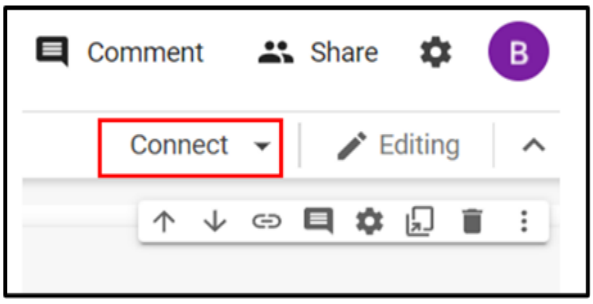



**Fig. 8.** *File connect*

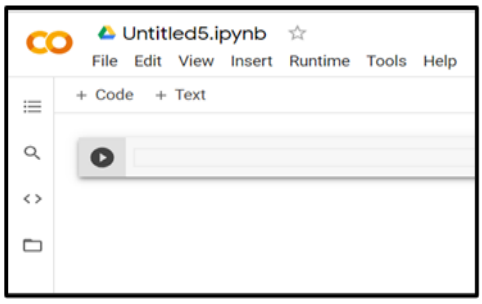



**Fig. 9.** *A cell with run button*

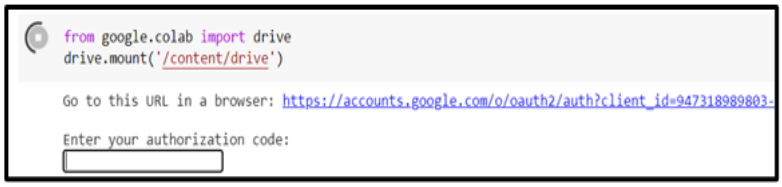



**Fig. 10.** *Google colab with Google drive authentication*

time. Attention was paid to this until the end of the epoch.

**Visualize and saving the model**

Plot history can also be used to visualise the model's output. The trained model was then saved to Google Drive as model1.hdf5, and the model inference was also shown. THen first pre-process the image dataset by resizing it, then generating the class indices.

**Prediction the dataset**

The Imread function must be used to upload any test dataset. After having given the cell, execute it. It will deliver the desired result.

**Testing**

A test was done to test the accuracy of the results of the classification of training data features with 100 maize FAW infestation data on each class in the research on corn plant FAW detection using the CNN approach, as shown in Table 3.

## RESULTS AND DISCUSSION

**Output of the model**

An algorithm was trained and generated the results at 10 epochs and 100 epochs shown in (Fig.11) and (Fig.12) by using photos of symptoms of fall armyworm infestation in maize as the training dataset. It manifests the training outcome for every 10 epochs and 100 epochs.

The training data set was able to achieve 100 per cent accuracy, whereas the validation accuracy was about 74 per cent on completion of 10 epochs during training the dataset. The visualisation of train and validation accuracy plots demonstrates that the model effectively identifies and distinguishes the symptoms caused by fall armyworm in maize (Fig.11a). Train and validation loss per epoch decreased as the number of epochs increased (Fig.11b).

The model was additionally trained across 100 epochs, ensuing in a training data set that was 100 per cent accurate and a validation data set that was roughly 87 per cent accurate (Fig.12). This implies that the trained model is neither overfitting nor underfitting. The model accuracy plots for both train and validation data are represented in Fig.12a. The model loss plots for both train and validation data are represented in Fig.12b. Training loss was only about 0.1 per cent, thus this model performs better and this implies that the trained model can perform accurate predictions based on images that will be fed to the system.

Lu *et al*. (2017) claimed that using their own convolutional neural network model, and they were able to achieve an accuracy of 95.48 per cent on a 500-leaf dataset of rice. Kulkarani (2018) reported that training data sets with 99.62 per cent accuracy of Mobile Net V2 model on the Plant Disease dataset, which is an open-source dataset that contains 54,306 photos of crop leaves classified into 38 different disease classifications. Ma et al. (2018) claimed that using their own convolutional neural network model, and they were able to achieve an accuracy of 93.4 per cent on 14,208 Cucumber leaf photos. Chen *et al.* (2019) claimed that using their own convolutional neural network model, and they were able to achieve an accuracy of 90.16 per

**Table 3.** Testing the image dataset

| Classification | Test dataset | True | False |
|---|---|---|---|
| Healthy maize leaves | 100 | 95 | 05 |
| Pin hole caused by FAW | 100 | 80 | 20 |
| Circular hole caused by FAW | 100 | 91 | 09 |
| Ragged hole caused by FAW | 100 | 98 | 02 |
| Whorl damage caused by FAW | 100 | 89 | 11 |
| Nil damage to slight damage at tip of the cobs | 100 | 91 | 09 |
| <25% of cob area showing FAW infestation | 100 | 83 | 17 |
| 26-50% of cob area showing FAW infestation | 100 | 87 | 13 |
| 51-75% of cob area showing FAW infestation | 100 | 94 | 06 |
| <75% of cob area showing FAW infestation | 100 | 96 | 04 |
| Healthy maize tassel | 100 | 98 | 02 |
| Fall armyworm infested maize tassel | 100 | 92 | 08 |
| Total | 1200 | 1094 | 106 |

```
Epoch 1/10
28/28 [==============================] - 1305s 45s/step - loss: 1.9683 - accuracy: 0.3686 - val_loss: 1.4026 - val_accuracy: 0.5790

Epoch 00001: val_loss improved from inf to 1.40255, saving model to /content/drive/MyDrive/Disease_Detection/best.hdf5
Epoch 2/10
28/28 [==============================] - 122s 4s/step - loss: 0.4452 - accuracy: 0.8772 - val_loss: 1.1976 - val_accuracy: 0.6121

Epoch 00002: val_loss improved from 1.40255 to 1.19755, saving model to /content/drive/MyDrive/Disease_Detection/best.hdf5
Epoch 3/10
28/28 [==============================] - 103s 4s/step - loss: 0.1350 - accuracy: 0.9624 - val_loss: 1.2949 - val_accuracy: 0.5974

Epoch 00003: val_loss did not improve from 1.19755
Epoch 4/10
28/28 [==============================] - 101s 4s/step - loss: 0.0566 - accuracy: 0.9868 - val_loss: 1.2512 - val_accuracy: 0.6213

Epoch 00004: val_loss did not improve from 1.19755
Epoch 5/10
28/28 [==============================] - 122s 4s/step - loss: 0.0258 - accuracy: 0.9957 - val_loss: 1.0525 - val_accuracy: 0.6857

Epoch 00005: val_loss improved from 1.19755 to 1.05246, saving model to /content/drive/MyDrive/Disease_Detection/best.hdf5
Epoch 6/10
28/28 [==============================] - 122s 4s/step - loss: 0.0153 - accuracy: 0.9980 - val_loss: 1.0503 - val_accuracy: 0.6967

Epoch 00006: val_loss improved from 1.05246 to 1.05026, saving model to /content/drive/MyDrive/Disease_Detection/best.hdf5
Epoch 7/10
28/28 [==============================] - 101s 4s/step - loss: 0.0114 - accuracy: 0.9991 - val_loss: 0.8761 - val_accuracy: 0.7445

Epoch 00007: val_loss improved from 1.05026 to 0.87606, saving model to /content/drive/MyDrive/Disease_Detection/best.hdf5
Epoch 8/10
28/28 [==============================] - 122s 4s/step - loss: 0.0107 - accuracy: 0.9981 - val_loss: 0.9911 - val_accuracy: 0.7188

Epoch 00008: val_loss did not improve from 0.87606
Epoch 9/10
28/28 [==============================] - 122s 4s/step - loss: 0.0039 - accuracy: 0.9997 - val_loss: 0.9514 - val_accuracy: 0.7537

Epoch 00009: val_loss did not improve from 0.87606
Epoch 10/10
28/28 [==============================] - 122s 4s/step - loss: 0.0035 - accuracy: 1.0000 - val_loss: 0.9713 - val_accuracy: 0.7482

Epoch 00010: val_loss did not improve from 0.87606
```

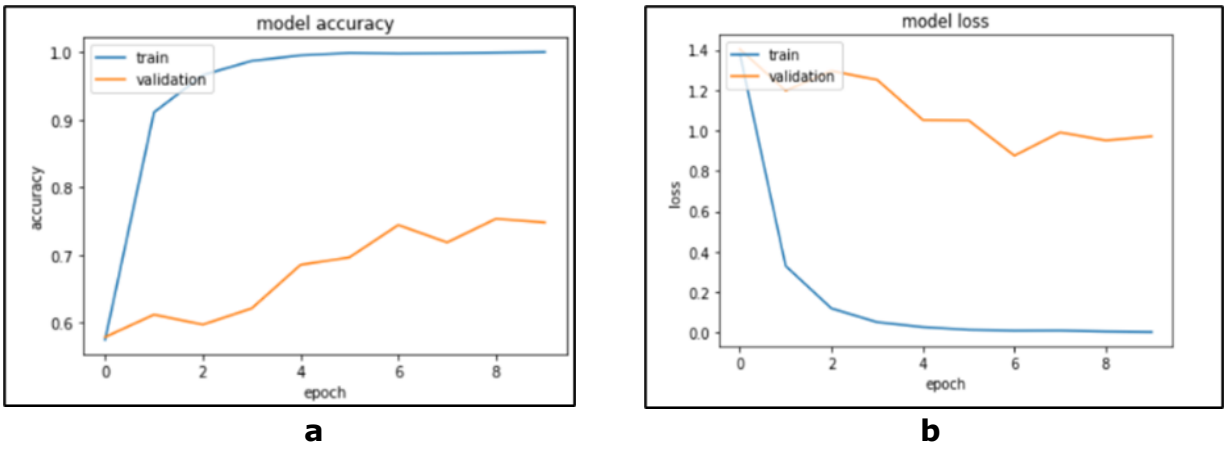



**Fig.11.** *Accuracy and loss for train and validation dataset at 10 epochs, a) Model accuracy plot for train and validation dataset. b) Model loss plot for train and validation dataset*

cent on 3810 tea leaf photos. The present study employed its own convolutional neural network model, which achieved 100% training accuracy on a dataset of 11,251 images of fall armyworm infestation maize divided into 12 classes as the present deep learning model performed well.

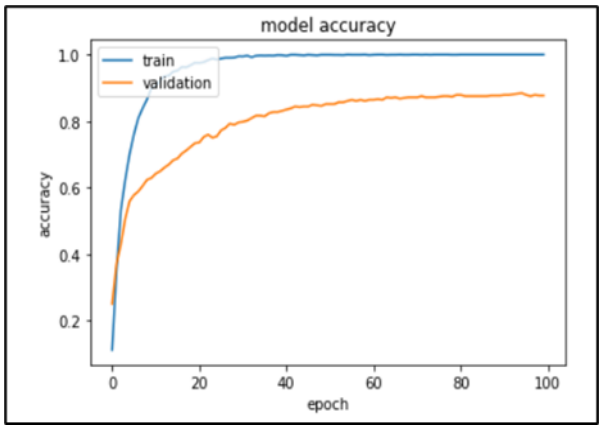
**Learning rate analysis**

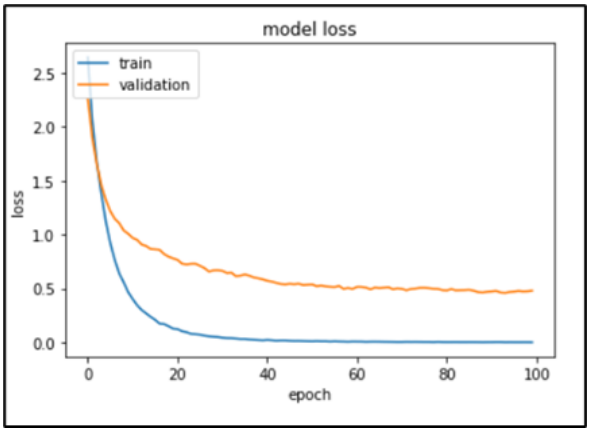
The learning rate is a hyper parameter. This reduces the model's inaccuracy by minimising the loss function. On a collection of 1088 ladies finger leaf image datasets, the exceptional accuracy of 94 percent was reached using CNN at a learning rate of 0.001 (Selvam and Kavitha, 2020). In this present study, at a learning rate of 0.00001, the outstanding accuracy of 100 per cent was reached on the classification of maize fall armyworm infestation. It was found that as the learning rate was increased or decreased beyond the ideal rate of 0.00001, the accuracy decreased.

**Development of mobile application**

Hidayat *et al*. (2019) used CNN to construct an android app via android studio for maize corn diseases, using a total dataset of 3854 pictures of diseases in corn plants, including three different forms of corn diseases including Common Rust, Gray Leaf Spot, and Northern

```
Epoch 90/100
28/28 [==============================] - 230s 8s/step - loss: 0.0019 - accuracy: 1.0000 - val_loss: 0.4690 - val_accuracy: 0.8768

Epoch 00090: val_loss did not improve from 0.46251
Epoch 91/100
28/28 [==============================] - 236s 8s/step - loss: 0.0021 - accuracy: 1.0000 - val_loss: 0.4723 - val_accuracy: 0.8787

Epoch 00091: val_loss did not improve from 0.46251
Epoch 92/100
28/28 [==============================] - 236s 8s/step - loss: 0.0032 - accuracy: 1.0000 - val_loss: 0.4770 - val_accuracy: 0.8787

Epoch 00092: val_loss did not improve from 0.46251
Epoch 93/100
28/28 [==============================] - 228s 8s/step - loss: 0.0043 - accuracy: 1.0000 - val_loss: 0.4624 - val_accuracy: 0.8805

Epoch 00093: val_loss improved from 0.46251 to 0.46243, saving model to /content/drive/MyDrive/Disease_Detection/best.hdf5
Epoch 94/100
28/28 [==============================] - 235s 8s/step - loss: 0.0015 - accuracy: 1.0000 - val_loss: 0.4588 - val_accuracy: 0.8824

Epoch 00094: val_loss improved from 0.46243 to 0.45881, saving model to /content/drive/MyDrive/Disease_Detection/best.hdf5
Epoch 95/100
28/28 [==============================] - 235s 8s/step - loss: 0.0020 - accuracy: 1.0000 - val_loss: 0.4672 - val_accuracy: 0.8842

Epoch 00095: val_loss did not improve from 0.45881
Epoch 96/100
28/28 [==============================] - 235s 8s/step - loss: 0.0015 - accuracy: 1.0000 - val_loss: 0.4709 - val_accuracy: 0.8787

Epoch 00096: val_loss did not improve from 0.45881
Epoch 97/100
28/28 [==============================] - 235s 8s/step - loss: 0.0019 - accuracy: 1.0000 - val_loss: 0.4767 - val_accuracy: 0.8750

Epoch 00097: val_loss did not improve from 0.45881
Epoch 98/100
28/28 [==============================] - 229s 8s/step - loss: 0.0018 - accuracy: 1.0000 - val_loss: 0.4721 - val_accuracy: 0.8787

Epoch 00098: val_loss did not improve from 0.45881
Epoch 99/100
28/28 [==============================] - 236s 8s/step - loss: 0.0017 - accuracy: 1.0000 - val_loss: 0.4737 - val_accuracy: 0.8768

Epoch 00099: val_loss did not improve from 0.45881
Epoch 100/100
28/28 [==============================] - 236s 8s/step - loss: 0.0011 - accuracy: 1.0000 - val_loss: 0.4795 - val_accuracy: 0.8768

Epoch 00100: val_loss did not improve from 0.45881
```




**a** **b**

**Fig.12.** *Accuracy and loss for train and validation dataset at 100 epochs. a) Model accuracy plot for train and validation dataset. b) Model loss plot for train and validation dataset*

Leaf Blight. The accuracy in recognising disease in corn plants is 99 per cent.

In my research, the android application (AIPES) was built using the trained model with the assistance of Android Studio. It was appended to the system as a library. App interface of the developed android application is shown in Fig.13. The fall armyworm infested maize can be identified in two possibilities available in the application. One is to select the file, and the other is to use the camera. Images can either uploaded from the gallery where the test data was stored , or a camera can be used to shoot field photos of fall armyworm infested photos. A maize leaf, cob and tassel sample screen from the mobile application that uses the test dataset to classify the fall armyworm symptom as seen in Fig.14, Fig.15, Fig.16 respectively. The image was fed into the classifier, which was then matched against the maize fall armyworm infection database to get the

relevant symptom classification description.

## Conclusion

The agricultural industry relies on early detection and recognition of these FAW infestations. In the present study, CNN was used to classify photos of maize affected by the autumn armyworm. Using images collected from maize research plots, the CNN architecture was trained. The present model had a training accuracy of 100% and a validation accuracy of 87% in terms of classification. This demonstrated that the CNN extracts essential information from photos with an unmanaged backdrop, which is necessary for classifying plant pest infestations. Experiments further revealed that the suggested CNN architecture could classify fall armyworm-infested maize into twelve categories. The number of datasets will be increased as part of this project's extension as the study can boost the high accuracy in the validation.

## Conflict of interest

The authors declare that they have no conflict of interest.

## REFERENCES


1. Alves, A.N., Souza, W. & Borges, D.L. (2020). Cotton pests' classification in field-based images using deep residual networks. *Computers and Electronics in Agriculture*, 174, 105488. https://doi.org/10.1016/j.compag.202 0.1 5488
2. Akhtar, A., Khanum, A., Khan, S.A. & Shaukat, A. (2013). Automated plant disease analysis (APDA): performance comparison of machine learning techniques. In *2013 11th International Conference on Frontiers of Information Technology*, 60-65. https://doi.org/10.1109/FIT.2013.19
3. Chen, J., Liu, Q. & Gao, L. (2019) Visual tea leaf disease recognition using a convolutional neural network model symmetry. https ://doi.org/10.3390/sym11030343.
4. Cheng, X., Youhua, Z., Yiqiong, C., Yunzhi, C. & YiYue, W. (2017). Pest identification via deep residual learning in complex background, *Computers and Electronics in Agriculture*, 141, 351-356.
5. Hidayat, U., Darusalam. & Irmawati, I. (2019). Detection of disease on corn plants using convolutional neural. *Journal of Computer Science and Information*, 12(1), 51-56. https://doi.org/10.21609/jiki.v12i1.695.
6. Huddar, S.R., Gowri, S., Keerthana, K., Vasanthi, S. & Rupanagudi, S.R. ( 2012). Novel algorithm for segmentation and automatic identification of pests on plants using image processing. In *2012 Third International Conference on Computing, Communication and Networking Technologies (ICCCNT'12)*,1-5. doi: 10.1109/ICCCNT.20 12.639 6012.
7. Kawasaki, Y., Uga, H., Kagiwada, S. & Iyatomi, H. (2015). Basic study of automated diagnosis of viral plant diseases using convolutional neural networks. In *International symposium on visual computing*, 638-645.
8. Kulkarni, O. (2018), August. Crop disease detection using deep learning. In *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*,1-4.
9. Nanni, L., Maguolo, G. & Pancino, F. (2020). Insect pest image detection and recognition based on bio-inspired methods. *Ecological Informatics*, *57*, 101089. https:// doi.org/10.1016/j.ecoinf.2020.101089.
10. Lu ,Y., Yi, S., Zeng, N., Liu, Y., Zhang, Y. (2017). Identifcation of rice diseases using deep convolutional neural networks. *Neurocomputing,* 267,378–384. https:// doi.org/10.1016/j.neucom.2017.06.023.
11. Ma, J. *et al* .(2018). A recognition method for cucumber diseases using leaf symptom images based on deep convolutional neural network. Comput Electron Agric 154:18–24. https://doi.org/10.1016/j. compag.2018.08.048.
12. Militante, S.V., Gerardo, B.D. & Dionisio, N.V. (2019). Plant leaf detection and disease recognition using deep learning. In *2019 IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE)* ,579-582. doi: 10.1109/ECICE47484.2019.8942686.
13. Murdia, L.K., Wadhwani, R., Wadhawan, N., Bajpai, P. & Shekhawat, S. (2016). Maize utilization in India: an overview. *American Journal of Food and Nutrition*, *4*(6),169-176.
14. Selvam, L. & Kavitha, P. (2020). Classification of ladies' finger plant leaf using deep learning. *Journal of Ambient Intelligence and Humanized Computing*,1-9.
15. Singh, V. & Misra, A.K. (2017). Detection of plant leaf diseases using image segmentation and soft computing techniques. *Information processing in Agriculture*, *4*(1), 41 -49. https://doi.org/10.1016/j.inpa.2016.10.005.
16. Sladojevic, S., Arsenovic, M., Anderla, A., Culibrk, D. & Stefanovic, D. (2016). Deep neural networks-based recognition of plant diseases by leaf image classification. *Computational intelligence and neuroscience*. https:// doi.org/10.1155/2016/3289801.
17. Syarief, M. & Setiawan, W. (2020). Convolutional neural network for maize leaf disease image classification. *Telkomnika*, *18*(3), 1376-1381. doi: 10.12928/ TELKOMNIKA.v18i3.14840.
18. Szegedy, C., Vanhoucke ,V., Ioffe, S., Shlens, J. & Wojna, Z. (2016). Rethinking the inception architecture for computer vision, *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2818-2826.
19. Wang, K., Zhang, S., Wang, Z., Liu, Z. & Yang, F. (2013). Mobile smart device-based vegetable disease and insect pest recognition method. *Intelligent Automation & Soft Computing*, *19*(3), 263-273. https://doi.org/10.1080/107 98587.2013.823783.
20. Yao, Q., Chen, G., Wang, Z., Zhang, C., Yang, B. & Tang, J. (2017). Automated detection and identification of white backed planthoppers in paddy fields using image processing. *Journal of Integrative Agriculture*, 16(7), 1547-1557. https://doi.org/10.1016/S2095-3119(16)61497-1.